WHAT IS CLAIMED IS:

1. A programmable processing system that executes multiple instruction contexts comprising:

an instruction memory for storing instructions that are executed by the system;

fetch logic for determining an address of an instruction, with the fetch logic comprising:

scheduling logic that schedules execution of the instruction contexts based on condition signals indicating an availability of a hardware resource, with the condition signals being divided into groups of condition signals, which are sampled in turn by the scheduling logic to provide a plurality of scan sets of sampled conditions.

2. The system of claim 1 wherein said scheduling logic further comprises:

control logic that determines when a context may be scheduled based on the execution of another context by said system.

3. The system of claim 2 wherein said scheduling logic further comprises:

a context store to store context information for a plurality of contexts, the context information including a priority field that indicates the execution priority level of each of the plurality of contexts; and

an executing contexts stack to indicate the priority level of a context marked for execution.

4. The system of claim 3 wherein the executing contexts stack has a first bit field that indicates a context having context information stored in the executing contexts stack is ready to pre-empt the currently executing context.

5. The system of claim 4 wherein said executing contexts stack has a second bit field that indicates a context having context information stored in said execution stack is currently being executed.

6. The system of claim 5 wherein the first bit field and the second bit field are used by said inhibit control logic to determine when a new context may be transferred to the executing contexts stack for pre-empting another context.

7. The system of claim 6 wherein the first bit field is set by the inhibit control logic when the new context is transferred to the executing contexts stack.

8. The system of claim 3 wherein said scheduling logic further comprises:

sampling logic that receives the condition signals; and

a scan set counter that selects a first scan set to be output from said sampling logic.

9. The system of claim 8 wherein said context store includes a starting event field associated with one of said plurality of contexts, with the starting event field used by said scheduling logic to determine whether a bit of the condition signal contained within the first scan set matches a bit indicated by the starting event field.

10. The system of claim 9 wherein the starting event field comprises:

a condition field indicating which condition signal within a scan set must be set for this context to be scheduled for execution; and

a scan set indicator field indicating which of the plurality of scan sets contains the condition signal which must be set for this context to be scheduled for execution.

11. The system of claim 10 wherein the starting event field further comprises:

a type bit field specifying the polarity of the condition signal that indicates the condition signal is set.

12. The system of claim 9 wherein said scheduling logic further comprises:

a low priority comparator and a high priority comparator that compare a selected scan set to a low priority start event context and a high priority start event context.

13. The system of claim 12 wherein said scheduling logic further comprises:

a scan word register that stores the last selected scan set and the last scan set counter value, said scan word register being input to said low priority comparator for comparing the last selected scan set at a time after the high priority comparison on the last context scan set has completed.

14. The system of claim 9 further comprise:

at least one input/output logic block, said input/output logic block also providing at least one of the condition signals to the scheduling logic.

15. The system of claim 9 wherein said inhibit control logic delays the enabling of the low priority context before the high priority context when an instruction causes a context exit.

16. The system of claim 9 further comprises:

a plurality of coprocessor engines that execute multiple instruction contexts, said plurality of coprocessor engines also providing a set of coprocessor condition signals to said scheduling logic.

17. The system of claim 14 wherein at least one said plurality of coprocessors further comprise:

at least one of a buffer and an address queue, said at least one coprocessor also providing at least one of the condition signals to the scheduling logic.

18. A method of operating a programmable processing system, the method comprising:

scheduling execution of an instruction context based on condition signals indicating an availability of a hardware resource, with the condition signals being divided into groups of condition signals, which are sampled to provide a plurality of scan sets of sampled conditions.

19. The method of claim 18 wherein context information is stored that includes a starting event for each context that may be scheduled for execution, the method further comprises:

determining that a starting event for a context matches one of the sampled conditions in one of the plurality of scan sets.

20. The method of claim 19 wherein said determining further comprises:

determining when a context may be scheduled based on a level of priority of a currently executing context.

21. The method of claim 20 further comprises:

dividing context information into at least two priority levels; and

scheduling a higher priority context for execution having a starting that matches a sampled condition before a lower priority context having the same starting event.

22. A computer program stored in a computer readable medium having instructions causing a computer to:

schedule execution of an instruction context based on condition signals based on an availability of a hardware resource, with the condition signals being divided into groups of condition signals, which are sampled to provide a plurality of scan sets of sampled conditions.

23. The computer program of claim 22 further comprising instruction causing a computer to:

determine that a starting event for a context matches one of the sampled conditions in one of the plurality of scan sets.